


## Research Article

# Lightweight Fine-Grained Multiowner Search over Encrypted Data in Cloud-Edge Computing

XueYan Liu <sup>1</sup>, LiJuan Huan,<sup>2</sup> RuiRui Sun,<sup>1</sup> and Jing Wang<sup>1</sup>

<sup>1</sup>*School of Computer Science and Engineering, Northwest Normal University, Lanzhou, China*

<sup>2</sup>*School of Mathematics and Statistics, Northwest Normal University, Lanzhou, China*

Correspondence should be addressed to XueYan Liu; liuxy@nwnu.edu.cn

Received 27 September 2022; Revised 19 December 2022; Accepted 20 December 2022; Published 6 January 2023

Academic Editor: Jie Cui

Copyright © 2023 XueYan Liu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Edge computing, as an extension of cloud computing, outsources encrypted sensitive data to edge nodes to decrease latency and improve broadband efficiency. Although attribute-based keyword search (ABKS) can ensure the security of outsourced data and promote fine-grained access control of data, deficiencies still exist under the general ABKS scheme in the cloud, as follows: shared files owned by a single data owner, inflexible access control, key escrow problem, and high computational overhead. We propose an attribute-based multidata owner searchable encryption scheme in cloud-edge computing with effective policy update to address these problems. The two-level access control is used to realize the common management and fine-grained sharing of data by the multidata owner. All user keys are jointly generated by the central authority and attribute authority, and a key escrow is no longer required. Moreover, partial encryption and decryption calculations are shifted from resource-constrained data owners and users to edge nodes. Furthermore, our scheme not only supports policy update but also realizes the dynamic updating of the index. The security proof and performance analysis show that the scheme has strong security and practicality in the edge computing environment.

## 1. Introduction

Cloud computing [1, 2] can provide services for massive resources and enable cloud clients to reduce the burden of local storage and computing. However, some new industries, such as smart cities, smart medicine, smart homes, and online education systems, in China start to emerge. Other smart devices, such as wearable medical devices, Internet of things (IoT) sensors, and smartphones, are also updated iteratively. The popularity of these applications poses a great challenge to centralised cloud computing, resulting in transmission delay and service degradation between users and the cloud. In particular, a large amount of data generated by IoT applications are usually stored in the cloud. Edge computing [3, 4] paradigm, which is an extension of cloud computing services to the network edge, has recently been studied to reduce delay and improve transmission efficiency. Edge computing is a distributed computing mode in which data storage is closer to the required location. It is

mainly or completely executed on distributed device nodes. Edge computing can provide various services for users with limited resources. The edge nodes are closer to the user than the cloud, as shown in Figure 1. When sensitive data are outsourced to honest but curious edge nodes, the data security and privacy issues [5] are still drawbacks that hinder the application of edge computing because data owners lose physical control over their data in edge nodes or in the cloud.

Encryption is a preferred method to protect data confidentiality and reduce data privacy leakage risks, but the retrieval of encrypted data becomes very difficult. The searchable encryption (SE) technology enables the user to securely and selectively retrieve the files containing the encrypted data according to the keywords specified by the user. In addition, access control on encrypted data is an essential function in practical applications. Attribute-based encryption (ABE) [6] is an effective option to provide fine-grained access control and realizes data security. Therefore, combining the advantages of the two aspects, attribute-based

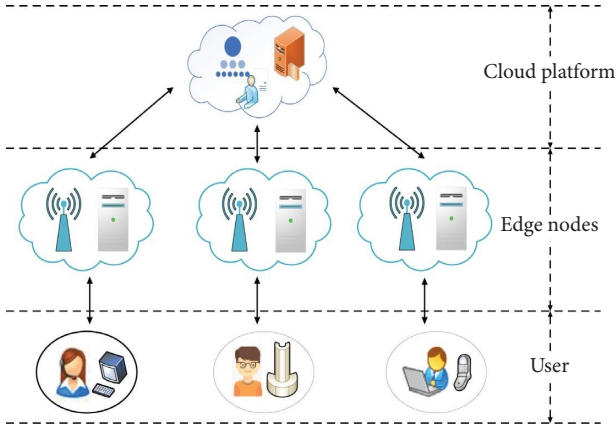


FIGURE 1: The infrastructure of edge computing.

keyword search (ABKS) [7–10] has received extensive attention in industrial and academic fields, such as online education. In the ABKS scheme, a user can access shared information if and only if the submitted token matches the index, and the attribute set satisfies the access policy embedded in the ciphertext.

ABKS is the most useful cryptographic tool for implementing fine-grained access control and keyword search functionalities, but most ABKS only consider the scenario of a single data owner, which is not suitable for many practical applications nowadays. In addition, there are also some issues to be concerned about: (1) the computing and storage overhead is proportional to the complexity of the access policy and the computing cost of the data owner and data user is too high. (2) When the attribute authority is maliciously attacked, the user's key is at risk of leakage. (3) Data owners cannot support the update of the access policy when the encrypted data are stored in the cloud.

Therefore, we design a lightweight and fine-grained multidata owner encryption scheme in cloud-edge computing to address existing issues and realize keyword search and policy update. The proposed scheme uses two-level access control to encrypt shared resources and generate index information to ensure data security. Miao et al. [11] also proposed a verifiable fine-grained keyword search scheme by multiple data owners in cloud environment, which realizes multidata owner sharing of resources. Compared with this scheme, our proposed scheme avoids key escrow, provides outsourcing encryption, and realizes policy update. Secondly, edge computing is introduced into our scheme. It allows edge computing to undertake part of the calculation, storage, and operation of search work, which not only reduces the computing burden of data owners and data users but also improves the transmission speed. In this way, the cloud server is also prevented from obtaining any information about the keywords submitted by the user, and the connection between the obtained file and the keyword cannot be inferred, thereby improving the privacy protection of the searchable encryption mechanism. Furthermore, compared with the previous CP-ABKS solution, our proposed scheme allows edge nodes to directly update the index information related to the access policy when the policy

changes. Specifically, the main contributions of this paper are as follows:

*1.1. Secure Multidata Owner Authorisation.* Shared data are co-owned by multidata owners, and a two-level access structure is used to control the user's access ability. The combination of these technologies strengthens authorization management and data access control.

*1.2. Fast Data Retrieval.* Edge nodes possess some storage and computing abilities; they can reduce the delay caused by transmission in cloud computing. Thus, the index is stored on edge nodes, and the search work is carried out on edge nodes, thereby improving the search efficiency.

*1.3. Lightweight Encryption and Decryption.* Most calculations of data encryption, index generation, and decryption are shifted to the edge nodes due to the introduction of edge computing, greatly reducing the amount of calculation of resource limited data owners and data users.

*1.4. Flexible Policy Update.* Dynamic policy update is supported in our scheme to enrich the expression ability of access policy and maintain the freshness of the algorithm, and it is only carried out by edge nodes.

The remainder of the paper is organized as follows: the literature review for the system is presented in Section 2. Section 3 introduces the necessary background information of the paper. Section 4 defines the system model and security analysis for the system. Section 5 presents our construction and proves its correctness. Section 6 analyzes the security of the scheme. In Section 7, the performance and experiment of the proposed method are analyzed. Section 8 concludes this paper.

## 2. Related Work

*2.1. Attribute-Based Searchable Encryption.* ABKS scheme has attracted much attention because of its fine-grained access control and search functions. Song et al. [12] first proposed the concept of searchable encryption in 2000 and implemented a searchable encryption scheme under the symmetric encryption system. Boneh et al. [13] proposed the first public key searchable encryption (PEKS) scheme in 2004, which can realize a single keyword search. In 2005, Park et al. [14] first proposed a multikeyword PEKS scheme. Kaushik et al. [15] combined attribute encryption and searchable encryption for the first time and proposed attribute-based keyword search (ABKS). Only data users whose attributes conform to the tree access structure can search for keywords. To enrich the functionality of ABKS scheme, Wang et al. [16] proposed a searchable and revocable encryption scheme based on multidata owner attributes, but the scheme did not realize the claimed multidata owner authorization function. Moreover, in most of the existing ABKS schemes, there are still some open problems that urgently need to be resolved, for example, the

decryption and encryption overhead is very high. It makes them impractical for many applications, for example, smart devices have limited resources. As a remedy for this problem, Green et al. [17] proposed the first content-based outsourcing decryption scheme, and the decryption process is safely transferred to the cloud server. From then on, large numbers of outsourcing schemes have been proposed to achieve lightweight, Ali and Sadeghi [18] proposed an attribute-based keyword search scheme in cloud computing and outsourced some encryption and decryption work to cloud servers. In 2021, Meng et al. [19] proposed an attribute-based dynamic keyword search encryption in fog computing and outsourced part of the encryption and decryption work to fog nodes. Miao et al. [20] proposed a lightweight encryption and fine-grained search scheme for data in fog computing, which outsources part of the encryption and decryption calculation to fog nodes and supports conjunctive keyword search and attribute update. Unfortunately, this scheme reduces the load on data owners and users, but the interaction between fog nodes and data users increases the transmission load. However, although these schemes realize keyword search and support outsourcing encryption and decryption, they do not satisfy the practical requirements in the actual situation to realize the sharing of data by multidata owners, and the computing burden of encryption and decryption in cloud/fog computing is very heavy. With the development of 5G and IoT technology, edge computing is regarded as a new data resource, which can provide many high-quality outsourcing services. In edge computing, enabling multidata owners to share data and reducing the computational overhead of encryption and decryption is the first challenge.

**2.2. Key Escrow.** In general attribute-based encryption schemes and ABKS schemes, such as schemes in [21–23], the keys of the data users are generated by attribute authority, so it has the decryption ability of the data users. However, when attribute authority is maliciously attacked, the key of the data user will be leaked, which makes the data owner's data face great challenges in terms of data privacy and confidentiality. Therefore, Liu et al. [24] proposed an anonymous hierarchical attribute encryption scheme in the electronic medical sharing environment, which introduces multiple authority attribute-based encryption technology to achieve fine-grained data access control and avoid the bottleneck of key escrow under a single authority. However, because multiple authorities have to communicate with each other, the system's performance is significantly reduced. To sum up, solving the problem of key generation and distribution and realizing key-free escrow is the second challenge.

**2.3. Policy Update.** In a practical application, when the data owner needs to update the access policy, he needs to fetch back the resourced data and reencrypt it with the new access policy, which easily causes large communication and computing overhead. Therefore, Sahai et al. [25] proposed a support dynamic certificate and ciphertext authorization

scheme. The idea of policy update is proposed for the first time, and the ciphertext is updated by the proxy method. Lai et al. [26] proposed an attribute-based ciphertext conversion scheme, in which the proxy server can update the access policy in the ciphertext without decryption. Li et al. [27] proposed a CP-ABE scheme in edge computing based on the CP-ABE scheme with dynamically updated access policies proposed by Yang et al. [28] and also provided an efficient online policy update method to manage attribute information. Besides, in schemes [29, 30], data owners can flexibly update access control policies and ciphertext. He will generate an update key and send it to the cloud. Using the update key, the cloud can update the access policy in the ciphertext to a new one. However, in these schemes, the issue of updating the index related to the new access policy is not concerned. Therefore, implementing policy updates and index updating is the third challenge.

### 3. Preliminaries

**3.1. Bilinear Pairing.** Let  $G$  and  $G_T$  to be two cyclic groups of prime order  $p$  and  $g$  is a generator of  $G$ . A bilinear map is a function  $e: G \times G \rightarrow G_T$  with the following properties:

- (1) Bilinearity:  $e(g^a, g^b) = e(g^b, g^a) = e(g, g)^{ab}$  for any  $a, b \in \mathbb{Z}_p$  and  $g \in G$
- (2) Nondegeneracy: there exists  $g \in G$  such that  $e(g, g) \neq 1$
- (3) Computability: there exists a polynomial-time algorithm to compute  $e(g, h)$ , for any  $g, h \in G$

**3.2. Decisional Bilinear Diffie–Hellman (DBDH) Assumption.** Given the tuple  $(A, B, C) = e(g^a, g^b, g^c)$ ,  $a, b, c, z \in_R \mathbb{Z}_p$ . The DBDH assumption is that no polynomial time (PPT) algorithm  $B$  is to be able to distinguish the tuple  $e(g, g)^{abc}$  from the tuple  $e(g, g)^z$  with negligible advantage.

**3.3. Decisional Parallel Diffie–Hellman (BDHE) Assumption.** The decisional  $q$ -parallel bilinear Diffie–Hellman exponent (BDHE) problem is that for any PPT algorithm, given  $\vec{y} = g, g^s, g^a, \dots, g^{a^q}, g^{a^{q+2}}, \dots, g^{a^{2q}}, \forall 1 \leq j \leq q, g^{s^{b_j}}, g^{a^j/b_j}, \dots, g^{a^q/b_j}, g^{a^{q+2}/b_j}, \dots, g^{a^{2q}/b_j}$ , and  $\forall 1 \leq j, k \leq q, k \neq j, g^{a \cdot s \cdot b_k/b_j}, \dots, g^{a^q \cdot s \cdot b_k/b_j}$ . It is difficult to distinguish  $(\vec{y}, e(g, g)^{a^{q+1} \cdot s})$  from  $(\vec{y}, z)$ , where  $g \in G, z \in G, a, s, b_1, \dots, b_q \in \mathbb{Z}_p$  are chosen independently and uniformly at random.

**3.4. Access Structure.** Let there be  $n$  attributes  $U = \{U_1, U_2, \dots, U_n\}$  in the system, and each attribute  $U_i (i \in [1, n])$  has a set of possible values  $V_i \in \{v_{i,1}, v_{i,2}, \dots, v_{i,n_i}\}$ . First, let  $\text{Att} = \{\text{Att}_1, \text{Att}_2, \dots, \text{Att}_n\}$  be a user attribute list, where  $\text{Att}_i \in V_i$ . Then, the access policy is represented as  $Q = \{Q_1, Q_2, \dots, Q_n\}$ , where  $Q_i \in V_i$ . If the attribute list  $\text{Att}$  matches with the access policy  $Q (\text{Att} = Q)$ , namely,  $\text{Att}_i \in Q_i$ , then the ciphertexts embedded with  $Q$  can be decrypted by the data user with  $\text{Att}$ .

## 4. System Model and Security Analysis

**4.1. System Model.** In this scheme, we consider a retrieval scenario in cloud-edge computing, which mainly involves six entities, namely, central authority (CA), attribute authority (AA), multidata owners (m-DOs), cloud server (CS), edge node (EN), and data user (DU), which are shown in Figure 2. Assuming that AA, CS, and EN are semihonest, CA is a fully trusted entity, and the specific functions of each entity are as follows:

- (1) Central authority (CA): CA generates the identity key for the user and the transformation public key and private key. It is also responsible for the registration of the data owner.
- (2) Attribute authority (AA): AA generates attribute keys for users.
- (3) Multidata owners (m-DOs): they work together to generate the ciphertext and index, respectively, using the access matrix and access policy.
- (4) Cloud server (CS): CS provides storage services. When a qualified query is received, the corresponding ciphertext will be returned.
- (5) Edge node (EN): EN has storage and computing capabilities, filling the gap between users and CS. It processes the ciphertext and index uploaded by m-DOs, reencrypts the ciphertext and index, and uploads the ciphertext to CS. Next, EN processes the search request of DU, retrieves the ciphertext from adjacent nodes and CS, predecrypts, and returns the results to DU. In addition, when the access policy is updated, EN updates the index.
- (6) Data user (DU): DU issues search queries according to his attribute private key. Moreover, DU makes a bit decryption computation to obtain shared data.

**4.2. Scheme Definition.** The system model includes seven stages, as follows:

- (1) System initialization phase:  
 $\text{Setup}(1^\lambda) \rightarrow (\text{PK}, \text{MSK})$ : given the security parameters  $\lambda$  and system attribute set  $U = \{U_1, U_2, \dots, U_n\}$ , outputs public key and master key of CA and AA, and public-private key pair of the m-DOs.
- (2) Key generation phase:  
 $\text{KeyCom}_{\text{CA}}(\text{MSK}_{\text{CA}}, \text{MSK}_{\text{AA}}, \mu) \rightarrow D$ : the algorithm is jointly executed by CA and AA to generate the user key. CA inputs the master key and authenticated the user, AA inputs secret value  $\mu$  and authenticated the user, and CA calculates a secret value  $D$ .  
 $\text{IssueKey}_{\text{CA}}(\text{IDU}) \rightarrow \text{SK}_{u,\text{CA}}$ : CA inputs the identity of the user IDU and outputs the identity transformation key of the user.

$\text{IssueKey}_{\text{AA}}(\mu, \text{Att}) \rightarrow \text{SK}_{u,\text{AA}}$ : AA inputs the random number  $\mu$  and the user's attribute set Att and outputs the user's attribute key.

- (3) Encryption phase:

$\text{Encrypt}_{\text{DOs}}(m, \text{PK}, (\text{PK}_o, \text{SK}_o), M_{d \times l}, kw_k, W) \rightarrow (\text{CT}_{\text{DOs}}, \text{Ind}_{\text{DOs}})$ : input the public-private key pairs, access matrix  $M_{d \times l}$ , keywords  $\{kw_k\}_{k \in W}$  and access policy  $W$ , and output the ciphertext  $\text{CT}_{\text{DOs}}$  and index  $\text{Ind}_{\text{DOs}}$ .

- (4) Outsourcing encryption phase:

$\text{Encrypt}_{\text{EN}}(\text{PK}, W, \text{CT}_{\text{DOs}}, \text{Ind}_{\text{DOs}}) \rightarrow (\text{CT}, \text{Ind})$ : input the ciphertext  $\text{CT}_{\text{DOs}}$ , index  $\text{Ind}_{\text{DOs}}$  and access policy  $W$ , and output the ciphertext CT and index Ind.

- (5) Retrieval phase:

$\text{TokenGen}(\text{Att}, kw', \text{SK}_{u,\text{AA}}) \rightarrow (\text{Tok})$ : DU runs the algorithm, inputs the user's private key  $\text{SK}_{u,\text{AA}}$ , attribute set Att and keywords  $kw'$ , and outputs the token Tok.

$\text{Search}(\text{Att}, \text{Ind}, \text{Tok}) \rightarrow \perp / \text{Ciphertext}$ : EN runs the algorithm, inputs index Ind, token Tok and the user's attribute set Att. If the user's attribute set satisfies the access policy and the verification equation is established, it continues; otherwise, it aborts.

- (6) Outsourcing decryption and the user decryption phase:

$\text{PartDec}(\text{CT}, \text{TPK}) \rightarrow \text{CT}_{\text{out}}$ : EN runs the algorithm, inputs ciphertext CT and transformation public key TPK, and outputs part of the ciphertext.  $\text{FinalDec}(\text{CT}_{\text{out}}, \text{TSK}) \rightarrow (K)$ : DU runs the algorithm, inputs part of the ciphertext  $\text{CT}_{\text{out}}$  and transformation private key TSK, and outputs plaintext  $m$ .

- (7) Policy update phase:

$\text{UpdatePolicyIndex}(W', \text{Ind}) \rightarrow \text{Ind}'$ : EN runs the algorithm, inputs the new access policy  $W'$ , and outputs the updated index  $\text{Ind}'$ .

## 4.3. Security Model

**4.3.1. Indistinguishability of Ciphertext.** The ciphertext indistinguishability security under chosen-plaintext attacks of a multidata owner ABKS scheme is defined by the game between a challenger  $C$  and a probabilistic polynomial time (PPT) adversary  $A$ :

**Setup:**  $C$  runs the Setup algorithm according to system parameters Param, outputs public key  $\text{PK}_{\text{CA}}$  of CA to  $A$ , and retains master secret key  $\text{MSK}_{\text{CA}}$ .

**Query Phase 1:**  $A$  submits attribute set Att with identity IDU to  $C$ ,  $C$  runs the key extraction query, and returns the private key to  $A$ .

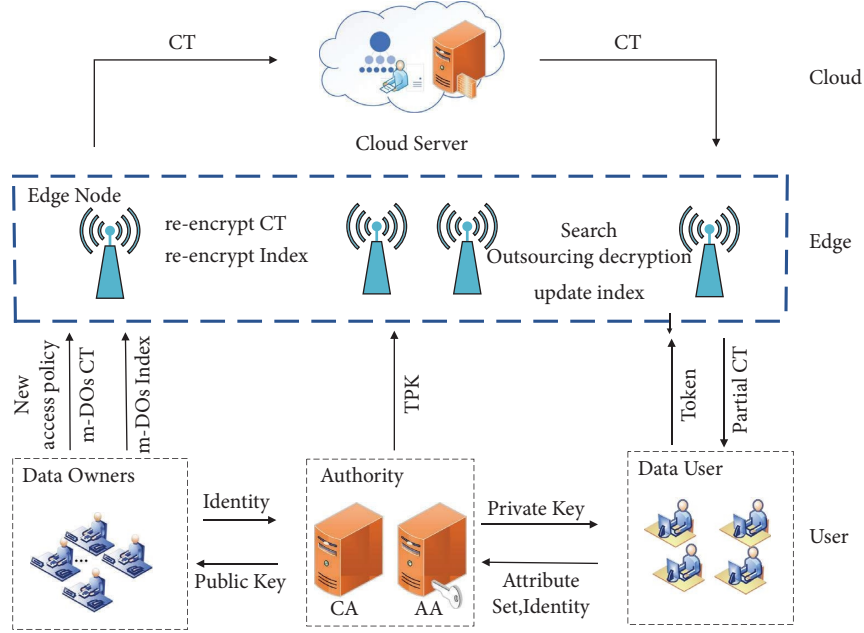


FIGURE 2: System model.

**Challenge:**  $A$  submits two messages  $K_0^*$  and  $K_1^*$  with the same length with access matrix to  $C$ .  $C$  randomly chooses a bit  $b \in (0, 1)$  and encrypts  $K_b^*$ . Finally,  $C$  sends the challenge ciphertext  $CT^*$  to  $A$ .

**Query Phase 2:**  $A$  continues to submit a series of attribute sets to  $C$  and still can make queries adaptively as in Query Phase 1.

**Guess:**  $A$  outputs a guess  $b' \in (0, 1)$  for  $b$  and if  $b' = b$ ,  $A$  wins the game. The advantage of the adversary for winning this confidentiality game is defined as  $\text{Adv}_A = |\Pr[b' = b] - 1/2|$ .

**Definition 1.** A multidata owner ABKS scheme without key escrow is secure under chosen-plaintext attacks, if the probability of the adversary  $A$  winning the game is negligible in a PPT.

**4.3.2. Chosen Keyword Indistinguishability.** The chosen keyword indistinguishability security of a multidata owner ABKS scheme is defined by the game between a challenger  $C$  and a PPT adversary  $A$ :

**Setup:**  $A$  selects the challenging access policy  $W_0, W_1$  and sends to challenger  $C$ .  $C$  runs Setup algorithm to return the public key  $PK_{AA}$  to  $A$  and retain the master secret key  $MSK_{AA}$ .

**Query Phase 1:**  $A$  adaptively selects the attribute set  $\text{Att}$  and queries to the following queries:

- (i) Private key query:  $C$  returns the associated private key to  $A$  by calling  $\text{KeyGen}$  if the attribute set  $\text{Att}$  satisfies access policy  $W_0, W_1$ .
- (ii) Token query:  $A$  submits attribute set  $\text{Att}$  and  $kw'$  to  $C$  for Token query. Then,  $C$  outputs the associated Token to  $A$  by calling  $\text{Token}$ .

**Challenge:**  $A$  submits two challenging keywords  $kw_0, kw_1$  with the same length, attribute set to  $C$ . If the attribute set satisfies the access policy  $W_0, W_1$  defined in query phase 1,  $A$  can get the access  $T_{kw'}$ . We define  $kw_0 = kw_1$ , and then  $C$  randomly selects  $b \in (0, 1)$  to generate the index ciphertext  $W_b$  and finally sends the index information to  $A$ .

**Query Phase 2:**  $A$  continues to issue private key queries and token queries, and if the keywords are not equal  $kw_0 \neq kw_1$ ,  $A$  cannot query the attribute set  $\text{Att}$  to satisfy the selected access policy  $W_0, W_1$ .

**Guess:**  $A$  outputs a guess  $b' \in (0, 1)$  for  $b$ . If  $b' = b$ ,  $A$  wins the game. The advantage of the adversary  $A$  for winning this confidentiality game is defined as  $\text{Adv}_A = |\Pr[b' = b] - 1/2| = \epsilon$ .

**Definition 2.** A multidata owner ABKS scheme without key escrow is secure under chosen keywords attacks, if the advantage  $A$  defined above for any PPT adversary is negligible.

In addition, this scheme also satisfies the security requirements of data privacy:

**Data privacy:** for each file shared by the data owners, they sign with public-private key pairs and encrypt the file with two-level access control. If and only if the first-level access control is met, second-level access control can be requested. This is particularly important to ensure the security and confidentiality of data.

## 5. Our Construction

With low latency and high efficiency provided by edge computing, resource-limited DU and m-DOs can outsource partial computational burden to EN, leaving a small part of the operations to be performed by themselves. Compared

with existing CP-ABKS schemes, our scheme can not only achieve fine-grained access control but also alleviate the computational burden on EN by adding a middle layer called EN. Thus, we consider the case of multiple owners where each file is co-owned by a group of DOs and give a general description for the system in Figure 3. In step (1), the authorities distribute the keys to m-DOs, DU, and EN. We refer to the scheme [31] and design a two-level access control over encrypted files with step (2). File encryption is based on a  $M_{d \times l}$  access matrix, in which each row of the matrix corresponds to an owner. Its purpose is to realize m-DOs authorization for DUs based on LSSS technology, the LSSS matrix has been illustrated in [32], which is the second-level access control for users. The first-level of access control occurs in the search phase. EN receives the ciphertext and index information sent by m-DOs and continues to re-encrypt them. EN saves the index information and uploads the ciphertext to CS finally. In order to filter users who want to obtain shared documents, m-DOs define access policies according to requirements and generate indexes embedded in access policies. DU must provide tokens embedded with their own attributes and identities for matching. Note that DU can request the second-level access control, if and only if, he satisfies the first-level access control. EN processes the search request of DU and retrieves the ciphertext from adjacent nodes and CS by step (3); after gaining the search results, EN first conducts the majority of operations to predecrypt with step (4) and returns partial ciphertext to DU. When the access policy used by m-DOs for encryption changes, which is shown by step (5). It is only necessary to send the new access policy to the ENs, and they will update the index. In the following, we show the main phase algorithms in the scheme, namely, system initialization phase, key generation phase, encryption and index generation phase, outsourcing encryption and index generation phase, retrieval phase, outsourcing decryption and user decryption phase, and policy update phase.

### 5.1. System Initialization Phase

- (i) **GlobalSetup**: on input the security parameters  $\lambda$  and the system attributes  $U = \{U_1, U_2, \dots, U_n\}$ , for each attribute value  $U_i$ , ( $i \in [1, n]$ ), there are a series of possible values  $V_i \in \{v_{i,1}, v_{i,2}, \dots, v_{i,n_i}\}$  and  $v_{i,j}$  ( $j \in [1, n_i]$ )  $\in Z_p^*$ .  $G$  and  $G_T$  are two groups of prime order  $p$  and  $g$  is a generator of  $G$ .  $e: G \times G \rightarrow G_T$  is the bilinear map, and  $H, H'$  is anti-collision hash functions  $H: \{0, 1\}^* \rightarrow Z_p^*$ ,  $H': \{0, 1\}^* \rightarrow G$ . Output system common parameters:

$$\text{Param} = (p, g, G, G_T, e, H, H'). \quad (1)$$

- (ii) **Setup<sub>CA</sub>**: CA selects  $\alpha \in Z_p^*$  randomly, then the master key and public key of CA is  $\text{MSK}_{\text{CA}} = \alpha, \text{PK}_{\text{CA}} = \{e(g, g)^\alpha, g^\alpha\}$ .
- (iii) **Setup<sub>AA</sub>**: AA selects  $x_{i,j}$  ( $i \in [1, n], j \in [1, n_i]$ ),  $\beta \in Z_p^*$  randomly and calculates  $A_{i,j} = g^{x_{i,j}}$ , and

generates the master key and public key of AA, where  $\text{MSK}_{\text{AA}} = (\beta, \{x_{i,j}\}_{i \in [1, n], j \in [1, n_i]})$ :

$$\text{PK}_{\text{AA}} = (g^\beta, \{A_{i,j}\}_{i \in [1, n], j \in [1, n_i]}). \quad (2)$$

- (iv) **Data Owner Registration**: m-DOs  $O = \{O_1, O_2, \dots, O_d\}$  send their identity information to CA, and CA generates public-private key pairs  $(\{PK_o, SK_o\}_{o \in [1, d]})$  for them.

### 5.2. Key Generation Phase

- (i) **KeyCom<sub>CA</sub>**: AA selects  $\mu \in Z_p$  randomly for authenticated users, which is unique to each user. AA and CA run two-party security protocols, AA inputs  $(\beta, \mu)$  and CA inputs  $\alpha$ . Under the condition that the two parties do not disclose any private information, CA obtains  $x_{\text{CA}} = \mu\alpha\beta$ . This is achieved by using two-party secure computing protocol [33] and can also be achieved by the construction in the literature [34]. CA randomly selects  $\gamma \in Z_p$ , then calculates  $Y = g^{Y_{\text{CA}}}$ , and sends  $Y$  to AA. After AA receives  $Y$ , it calculates  $T = Y^{1/\beta}$  and returns it to CA. Finally, CA calculates  $D = T^{1/\gamma} = g^{\mu\alpha}$ .
- (ii) **IssueKey<sub>CA</sub>**: CA outputs the user's identity key for identified user IDU. CA randomly selects  $z \in Z_p^*$  as the transformation private key and calculates the transformation public key  $k_o = g^z$ ,  $k_1 = g^{1+\mu\alpha z/H'(\text{IDU})}$ ,  $k_2 = g^{z\mu\alpha + H'(\text{IDU})}$ . Last outputs user identity transformation key is  $\text{TSK} = z$ ,  $\text{TPK} = \{k_o, k_1, k_2\}$ ,  $\text{TPK} = \{k_o, k_1, k_2\}$ .
- (iii) **IssueKey<sub>AA</sub>**: AA is based on the random number  $\mu$  selected in **KeyCom<sub>CA</sub>** and inputs the user's attribute set  $\text{Att} = \{\text{Att}_1, \text{Att}_2, \dots, \text{Att}_n\}$ . Then, it selects  $r \in Z_p$  randomly and output the attribute key:

$$\text{SK}_{u, \text{AA}} = (k_{3,i} = g^{r\beta\mu H'(\text{Att}_i)}, k_4 = g^{r\mu \sum_{i=1}^n x_{i,j}}). \quad (3)$$

Finally, the user can get his whole key  $\text{SK} = (\text{TSK}, \{k_{3,i}\}_{i \in [1, n], j \in [1, n_i]}, k_4)$ , and  $\text{TPK}$  is uploaded to the edge node.

**5.3. Encryption and Index Generation Phase.** Given file  $F = \{m\}$  and keywords  $\{kw_k\}_{k \in W_\varphi}$ , m-DOs generated file ciphertext and index:

- (i) **Encrypt<sub>DOs</sub>**: m-DOs first generate identity signatures  $\sigma_i, i = 1, 2, \dots, d$  for files  $F$  using public-private key pairs  $(PK_o, SK_o)$ . For file  $m \in F$ , m-DOs first encrypt the file  $m$  with a symmetric encryption key  $K \in G_T^*$  as  $C_m = \text{Enc}_K(m)$ . Then, m-DOs encrypt the symmetric key  $K$  with the attribute cryptography technology.  $M_{d \times l}$  is a matrix, a function  $\rho(i)$  maps each row  $M$  of  $M_{d \times l}$  to a DO. m-DOs select a column vector  $\vec{v} = (s, r_2, r_3, \dots, r_l) \in_R Z_p^l$ , compute  $\lambda_i = \vec{M}_i \vec{v}_i, i = 1, 2, \dots, l$ , and  $s$  is a secret value. We set ciphertext:

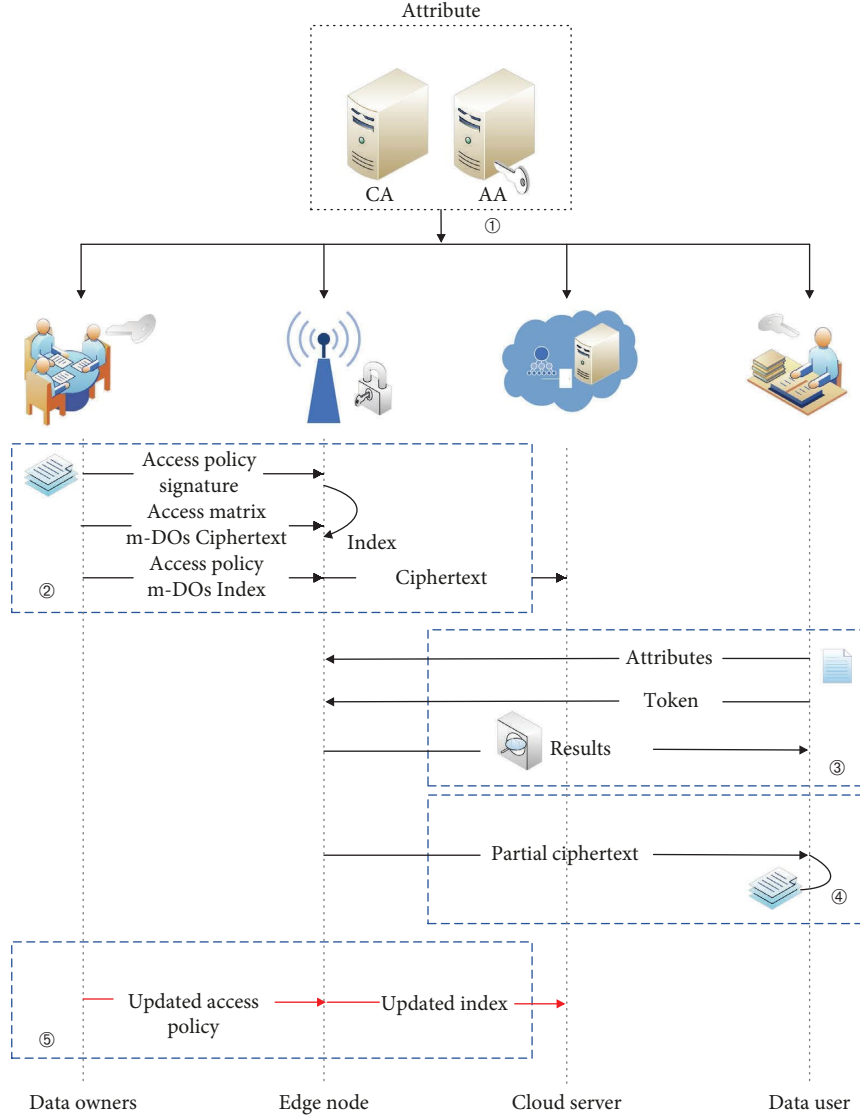


FIGURE 3: Framework of the system.

$$\begin{aligned}
 C_0 &= Ke(g, g)^{\alpha s \sum_{i=1}^d \sigma_i}, \\
 C_{1,i} &= g^{\lambda_i \alpha \sigma_i}, \\
 C' &= g^s.
 \end{aligned} \tag{4}$$

m-DOs output ciphertext:

$$CT_{DOs} = (C_m, C_0, C', \{C_{1,i}, \sigma_i\}_{i=1,2,\dots,d}). \tag{5}$$

- (ii) **Index<sub>DOs</sub>**: m-DOs define access policies  $W = \{W_1, W_2, \dots, W_n\}$  firstly. For keywords  $\{kw_k\}_{k \in W_\varphi}$ , m-DOs select  $\eta_i \in Z_p$  and compute  $\eta = \sum_{i=1}^d \eta_i$ . We set index  $I_1 = g^{\alpha \eta / \sum_{k \in W_\varphi} H(kw_k)}$ ,  $I' = g^{\beta \eta}$ . Output index  $Ind_{DOs} = \{W, I_1, I'\}$ . Finally, we upload  $CT_{DOs}$  and  $Ind_{DOs}$  to the edge node.

#### 5.4. Outsourcing Encryption and Index Generation Phase

- (i) **Encrypt<sub>EN</sub>**: inputs public key, ciphertext  $CT_{DOs}$ , signatures  $\sigma_i$ , and EN selects  $c_i \in Z_p^*$  randomly. We set ciphertext:

$$\begin{aligned}
 C_2 &= C' \sum_{i=1}^d c_i \sigma_i \\
 &= g^s \sum_{i=1}^d c_i \sigma_i, \\
 C_3 &= \prod_{i=1}^d C'^{c_i \sigma_i H' (IDO)} \\
 &= g^s \sum_{i=1}^d c_i \sigma_i H' (IDO).
 \end{aligned} \tag{6}$$

Output ciphertext:

$$CT = (C_m, C_0, C', \{C_{1,i}, \sigma_i\}_{i=1,2,\dots,d}, C_2, C_3). \tag{7}$$



- (ii) **Index<sub>EN</sub>**: EN runs the algorithm and inputs index  $Ind_{DOs}$ . According to the defined access policy  $W = \{W_1, W_2, \dots, W_n\}$  and  $W_i \in V_i (i \in [1, n])$ . For each  $W_i = v_{i,j}$ , EN selects  $\tau_i \in Z_p^*$  randomly and sets

$$I_2 = \prod_{i=1}^n A_{i,j}^{\tau_i} = g^{\sum_{i=1}^n x_{i,j} \tau_i}, \quad (8)$$

$$I_3 = g^{\beta \sum_{i=1}^n \tau_i H'(W_i)}.$$

Output index  $Ind = \{W, I_1, I', I_2, I_3\}$ .

Finally, we upload  $CT$  to  $CS$  for storage and save  $Ind$  by themselves.

### 5.5. Retrieval Phase

- (i) **TokenGen**: let  $\{kw_k'\}_{k \in W}$  be keywords set, DU with attribute sets  $Att$  first selects  $e \in Z_p$  randomly and sets token  $T_{1,k} = g^{\beta e H(kw_k')}$ ,  $T_2 = g^{\alpha e}$ ,

$$T_3 = \prod_{i=1}^n k_{3,i}^e$$

$$= g^{r\mu\beta e \sum_{i=1}^n H'(Att_i)}, \quad (9)$$

$$T_4 = k_4^e$$

$$= g^{re\mu \sum_{i=1}^n x_{i,j}}.$$

Output token  $Tok = (\{T_{1,k}\}_{k \in W}, T_2, T_3, T_4)$ .

- (ii) **Search**: inputs user's attribute set  $Att$ , access policy  $W$ , index and token. First, EN checks whether the user's attribute set satisfies access policy. If not, the algorithm aborts. Otherwise, we judge whether the following equation holds  $e(I_2, T_3)e(I', T_2)/e(T_4, I_3) = e(I_1, \prod_{k \in W} T_{1,k})$ . If the equation holds, we output the storage address, otherwise abort.

### 5.6. Outsourcing Decryption and User Decryption Phase

- (i) **PartDec**: Suppose  $P$  is a matrix access structure and  $S \in P$  is an authorization set with  $I = \{i: \rho(i) \in S\} \subset (1, 2, \dots, d)$ , there are a set of constant  $\omega_i$  which makes the equation  $\sum_{i \in I} \omega_i \bar{M}_i = (1, 0, \dots, 0)$  hold and  $\sum_{i \in I} \omega_i \lambda_i = s$ . EN computes  $CT_{out} = e(k_2, C_2)/e(k_1, C_3)e(\prod_{i \in I} C_{1,i}^{\omega_i}, k_0)$ . Finally, outputs partial ciphertext  $CT_{out}$  and sends it to DU.
- (ii) **FinalDec**: DU performs simple calculation  $K = C_0 CT_{out}^{1/z}$  by using the transformation private key TSK to obtain symmetric key  $K$  and gets  $m = Dec_K(C_m)$ .

### 5.7. Policy Update Phase

**5.7.1. UPdatepolicyIndes**. When the access policy is updated, m-Dos only need to send the new access policy  $W'$  to EN, and EN updates corresponding index.

First, the EN runs algorithm PolicyCompare and compares the old and new policies. Output new index information and store it in  $Ind_1, Ind_2, Ind_3$ . Let  $n/n'$  represent the number of attributes in the old/new access policy  $W/W'$ , respectively. There are three cases:

- (a) When the number of attributes in the access policy  $W'$  is reduced to  $n'$ , and  $n' < n$ . EN updates the index as follows:

$$I_2' = \frac{I_2}{\prod_{i=1}^n g^{x_{i,j} \tau_i}},$$

$$I_3' = \frac{I_3}{g^{\beta \sum_{i=1}^n \tau_i H'(W_i)}}, \quad (10)$$

$(j, i) \in Ind_1$ .

Then, updated index information is stored in the  $Ind_1$ .

- (b) When the number of attributes in the access policy  $W'$  is added to  $n'$ , and  $n' > n$ . EN updates the index as follows:

$$I_2' = I_2 \prod_{i=1}^{n'} g^{x_{i,j} \tau_i},$$

$$I_3' = I_3 g^{\beta \sum_{i=1}^{n'} \tau_i H'(W_i)} \quad (j, i) \in Ind_2. \quad (11)$$

Then, updated index information is stored in the  $Ind_2$ .

- (c) In the access policy  $W'$ , when the number of attributes changes in part and remains unchanged in part, EN randomly selects  $\tau_j' \in Z_p^*$  and updates the index as follows:

$$I_2' = \prod_{i=1}^{n'} g^{x_{i,j} \tau_i} g^{x_{i,j'} \tau_j'},$$

$$I_3' = \prod_{i=1}^{n'} g^{\beta \tau_i H'(W_i)} g^{\beta \tau_j' H'(W_i)} \quad (j, i) \in Ind_3. \quad (12)$$

Then, updated index information is stored in the  $Ind_3$ .

Output updated index  $Ind' = \{I_1, I', I_2', I_3'\}$ .

In this process, m-DOs only need to send a new access policy to EN, and EN uses the new access policy to update the index. Therefore, it reduces the calculation burden of DOs and ensures the security of the index.



## 5.8. Correctness

Correctness of the search phase:

$$\begin{aligned}
\frac{e(I_2, T_3)e(I', T_2)}{e(T_4, I_3)} &= \frac{e\left(g^{\sum_{i=1}^n x_{i,j} \tau_i}, g^{r\mu\beta e \sum_{i=1}^n H'(Att_i)}\right) e(g^{\beta\eta}, g^{ae})}{e\left(g^{re\mu \sum_{i=1}^n x_{i,j}}, g^{\beta \sum_{i=1}^n \tau_i H'(W_i)}\right)} \\
&= \frac{e(g, g)^{r\mu\beta e \sum_{i=1}^n x_{i,j} \tau_i \sum_{i=1}^n H'(Att_i)} e(g, g)^{\beta\eta ae}}{e(g, g)^{r\beta\mu e \sum_{i=1}^n x_{i,j} \sum_{i=1}^n \tau_i H'(W_i)}} \\
&= e\left(g^{a\eta / \sum_{k \in W_\varphi} H(kw_k)}, \prod_{k \in W_\varphi} g^{\beta e H(kw'_k)}\right) \\
&= e\left(I_1, \prod_{k \in W_\varphi} T_{1,k}\right).
\end{aligned} \tag{13}$$

Correctness of the part decryption phase:

$$\begin{aligned}
CT_{out} &= \frac{e(k_2, C_2)}{e(k_1, C_3) e\left(\prod_{i \in I} C_{1,i}^{\omega_i}, k_0\right)} \\
&= \frac{e\left(g^{\mu\alpha z + H'(IDU)}, g^s \sum_{i=1}^d c_i \sigma_i\right)}{e\left(g^{1+\mu\alpha z/H'(IDU)}, g^s \sum_{i=1}^d c_i \sigma_i H'(IDO)\right) e\left(g^{\sum_{i \in I} \lambda_i \omega_i \alpha \sigma_i}, g^z\right)} \\
&= \frac{1}{e(g, g)^{sz\alpha \sum_{i=1}^d \sigma_i}}.
\end{aligned} \tag{14}$$

Correctness of the user decryption phase:

$$\begin{aligned}
K &= C_0 CT_{out}^{1/z} \\
&= Ke(g, g)^{\alpha s \sum_{i=1}^d \sigma_i} \frac{1}{e(g, g)^{1/z z \alpha \sum_{i=1}^d \sigma_i}}.
\end{aligned} \tag{15}$$

## 6. Security Analysis

**Theorem 1.** Under the assumption of  $q$ -Parallel BDHE, the multidata owner ABKS scheme without key escrow is secure against indistinguishable chosen-plaintext attacks.

*Proof.* Assume that there is a PPT adversary  $A$  who can win the ciphertext indistinguishability security game with nonnegligible advantage  $\varepsilon$ :

Setup:  $C$  runs the Setup algorithm according to system common parameters Param, outputs public key  $PK_{CA}$  to  $A$ , and retains master secret key  $MSK_{CA}$ .

Query Phase 1:  $A$  queries the key to the challenger  $C$ . The key generated by the two-party secure computing protocol is secure and will not disclose private information.  $A$  submits attribute set Att with identity IDU to  $C$ ,  $C$  randomly selects  $\mu^*$  running algorithm  $KeyCom_{CA}$ , sets identity key  $k_1 = g^{1+\mu\alpha z/H'(IDU)}$ ,  $k_2 = g^{z\mu\alpha + H'(IDU)}$ , and returns the private key to  $A$ .

Challenge:  $A$  submits two messages  $K_0^*$  and  $K_1^*$  with the same length with access matrix to  $C$ .  $C$  randomly chooses a bit  $b \in (0, 1)$  and sets  $C_0^* = K_b^* e(g, g)^{\alpha s \sum_{i=1}^d \sigma_i}$  and randomly selects vectors  $\vec{v} = (s, r'_2, r'_3, \dots, r'_l) \in_R Z_p^l$  that are calculated by sharing secret value  $s$  and computes  $C_{1,i}^* = g^{\lambda_i \alpha \sigma_i}$ ,  $C_l^* = g^s$ . Finally,  $C$  sends the challenge ciphertext  $CT^*$  to  $A$ . Note that the security of the shared challenge ciphertext  $C_{1,i}^*, C_l^*$  is guaranteed by the embedded access matrix information.

Query Phase 2:  $A$  makes queries adaptively as in Query Phase 1.

Guess:  $A$  outputs a guess  $b' \in (0, 1)$  for  $b$  and if  $b' = b$ ,  $A$  wins the game. In other words, it is effective to  $CT^*$  under the assumption of  $q$ -Parallel BDHE. The advantage of the adversary for winning this confidentiality game is defined as  $\text{Adv}_A = |\Pr[b' = b] - 1/2|$ . Then, the scheme is proved to achieve selective plaintext security.  $\square$

**Theorem 2.** *The multidata owner ABKS scheme without key escrow is secure under chosen keyword attack if the DBDH problem is hard.*

*Proof.*  $A$  defines a challenging access policy  $W_0, W_1$  before the system establishment stage. In the security game of nonselective identity,  $A$  can submit an attribute set  $\text{Att}$  and satisfy two access strategies  $W_0, W_1$  at the same time, and then  $A$  can obtain the corresponding search results. During this period, in addition to the returned search results,  $A$  cannot obtain information about the access policy  $W_0, W_1$ , which corresponds to the access policy to be updated later:

Setup:  $A$  selects the challenging access policy  $W_0, W_1$  and sends to  $C$ .  $C$  runs Setup algorithm to return public key  $\text{PK}_{AA}$  to  $A$  and retain master secret key  $\text{MSK}_{AA}$ .

Query Phase 1:  $A$  adaptively selects the attribute set  $\text{Att}$  and queries to the following queries:

- (i) Private key query: if the attribute set  $\text{Att}$  satisfies the selected access policy  $W_0, W_1$  and  $C$  runs the key generation algorithm. Under the two-party security protocol,  $AA$  and  $CA$  randomly select  $r \in Z_p$  to jointly generate user attribute private key  $k_{3,i} = g^{r\beta\mu H'(\text{Att}_i)}$ ,  $k_4 = g^{r\mu \sum_{i=1}^n x_{i,j}}$  and send it to  $A$ .
- (ii) Token query:  $A$  submits attribute set  $\text{Att}$  and  $kw'$  to  $C$  for token query. Then,  $C$  randomly selects  $e \in Z_p$ , sets token

$$\begin{aligned} T_{1,k} &= g^{\beta e H(kw'_k)}, \\ T_2 &= g^{ae}, \\ T_3 &= g^{r\mu\beta e \sum_{i=1}^n H'(\text{Att}_i)}, \\ T_4 &= g^{r\mu e \sum_{i=1}^n x_{i,j}} \text{ to } A. \end{aligned} \quad (16)$$

Challenge:  $A$  submits two challenging keywords  $kw_0, kw_1$  with the same length, attribute set to  $C$ . If the attribute set satisfies the access policy  $W_0, W_1$  defined in query phase 1,  $A$  can get the access  $T_{kw'}$ . We define  $kw_0 \neq kw_1$ , and then  $C$  randomly selects  $b \in (0, 1)$  to set token  $T_{1,b} = g^{\beta e H(kw'_b)}$ ,  $T_2 = g^{ae}$ ,  $T_3 = g^{r\mu\beta e \sum_{i=1}^n H'(\text{Att}_i)}$ , and  $T_4 = g^{r\mu e \sum_{i=1}^n x_{i,j}}$  and finally sends the token information to  $A$ .

Query Phase 2:  $A$  continues to issue private key queries and token queries, If the keywords are not equal  $kw_0 \neq kw_1$ ,  $A$  cannot query the attribute set  $\text{Att}$  to satisfy the selected access policy  $W_0, W_1$ .

Guess:  $A$  outputs a guess  $b' \in (0, 1)$  for  $b$ . If  $b' = b$ ,  $A$  wins the game. Outputs  $b' = b = 1/2 + \varepsilon$ . Otherwise,  $A$  outputs  $b' = b = 1/2 + \varepsilon$ . Thus, the adversary  $A$  of solving the DBDH problem is

$$\begin{aligned} \left| \frac{1}{2} \Pr[b' = b] + \frac{1}{2} \Pr[b' \neq b] \right| &= \left| \left[ \frac{1}{2} \left( \frac{1}{2} + \varepsilon \right) + \frac{1}{2} \cdot \frac{1}{2} \right] - \frac{1}{2} \right| \\ &= \frac{\varepsilon}{2}. \end{aligned} \quad (17)$$

In addition, our scheme also satisfies the security and privacy requirements of data privacy under the assumption of DBDH.

Data privacy: first, we encrypt sensitive data to prevent data leakage. Secondly, we use two layers of access control to prevent unauthorized users from leaking secrets (the first layer is to filter users' identities and attribute sets, and the second layer is to satisfy the authorization of multiple data owners). Thirdly, although outsourcing decryption is adopted, the most critical calculations are completed by users to prevent EN from leaking secrets.  $\square$

## 7. Performance Analysis

### 7.1. Theoretical Analysis

**7.1.1. Function Comparison.** Table 1 shows the comparison between our scheme and related schemes [29, 31, 34–36] in terms of the number of data owners, policy updates, without key escrow, computing outsourcing, and searchability. Our scheme realizes multikeyword search; schemes [31, 35] only support single keyword search, whereas schemes [29, 34, 36] do not provide a search function. Our scheme and scheme [31] realize the multidata owner authorization management of shared files, but scheme [31] does not discuss access policy updating. Our scheme avoids the problem of key escrow, as well as scheme [34]. However, other schemes fail to solve this problem. In terms of computing outsourcing, our scheme introduces edge computing, so most encryption and decryption works are outsourced to edge nodes, whereas the data owners and users only perform few calculations. In scheme [35], only the decryption is outsourced to the cloud, and the encryption calculation is still performed by the data owner. However, other schemes do not consider outsourcing calculations. Therefore, our scheme is functional.

**7.1.2. The Storage Cost.** In this subsection, a storage cost comparison between our scheme and related schemes [29, 31, 35, 36] is presented in terms of the public key, transformation key, DO side ciphertext, cloud/fog/edge side ciphertext, side ciphertext, and token storage size, as shown in Table 2. The public key, transformation key, DO side ciphertext, and cloud/fog/edge side ciphertext occupy less storage space in our scheme. In terms of token storage, our scheme is independent of the number of attributes and related to the number of keywords, whereas the token

TABLE 1: Function comparison.

	Multiple data owners	Policy update	Without key escrow	Searchable	Computing outsourcing
[29]	×	√	×	×	Encryption/decryption
[31]	√	×	×	Single keyword	×
[34]	×	×	√	×	×
[35]	×	×	×	Single keyword	Decryption
[36]	×	√	×	×	Encryption/decryption
Ours	√	√	√	Multiple keyword	Encryption/decryption

storage in schemes [31, 35] increases linearly with the increase in the number of attributes. By contrast, our scheme has less storage overhead.

**7.1.3. The Computational Cost.** The computational overhead mainly involves bilinear pairing and modular exponential operations. Let  $|U|$  represent the number of system attributes,  $|G|/|G_T|$  represents the size of the element in the  $G/G_T$ ,  $n$  indicates the number of the user attributes,  $n_s$  represents the number of keywords used in token generation,  $l$  represents the number of attributes in the access policy, and  $d$  represents the number of data owners. Respectively,  $E_G/E_{G_T}, E_P$  represent the exponential operation in  $G/G_T$  and pairing operation.

For convenience, as shown in Table 3, we divide the encryption stage into two parts: DO encryption and CS/Fog/EN encryption. In the DO encryption stage, DO has completed all encryption operations in schemes [31, 35]. Thus, the amount of calculation is larger than that of other schemes. In schemes [29, 36] and our scheme, part of the encryption work has been outsourced from a third party, so the calculation load of DOs is relatively low. In our scheme, edge computing is introduced; the amount of calculation at the DO end is  $(d + 1)E_G + E_P$ , and the calculation amount at the EN end is  $2E_G$ . Similarly, the decryption phase is divided into two parts: CS/Fog/EN decryption and the user decryption. In these schemes, although the user decryption computation is one  $E_P$  operation, the outsourcing decryption computation is evidently different. The amount of computation required by EN to predecrypt the ciphertext is  $3E_P$  in our scheme, whereas the amounts of computation in scheme [29, 36] are  $2E_P$  and  $3E_P$ , respectively. In schemes [31, 36], the decryption is completed by the user. Therefore, the overall encryption/decryption computation of our scheme is relatively low.

In the search stage, the amount of token calculation in schemes [31, 35] increases linearly with the increase in the number of attributes and only realizes a single keyword search. If these schemes perform multiple keyword searches, then the calculation amount is also related to the number of keywords. However, in our scheme, the token generation is only related to the number of keywords, and multiple keyword search is implemented. In particular, the search work in our scheme is carried out by EN, and the amount of

calculation is  $4E_P$ , which is smaller than those in schemes [31, 35].

The theoretical analysis indicates that the calculation amount of our scheme is relatively low in the data user side (DO and user) and the outsourcing end.

**7.2. Experimental Simulation.** To further evaluate the performance of our scheme, we conducted a series of simulation experiments. Experiments are implemented on a platform Windows 10 with 2.70 GHz Intel (R) Core i5-7200u CPU, 8 GB RAM by using pairing-based cryptography (PBC) [37] with a large prime of 512 bits.

Figure 4 shows the computational cost of *encryption on DO side*, as well as the *outsourcing encryption and decryption*. The experimental simulation diagram is no longer provided here given that the decryption cost of the user is the same. Figure 4(a) shows that the computational cost in schemes [29, 31, 35, 36] increases linearly with the number of attributes, whereas our scheme is independent of the number of attributes, which means that the time cost remains constant and is lower than that in other schemes. Figures 4(b) and 4(c) introduce the comparison of computational cost in *outsourcing encryption* and *outsourcing decryption*, respectively. The computational cost of outsourcing encryption and decryption time in schemes [29, 36] continues to increase linearly with the increase in the number of attributes, whereas the computational cost of our scheme is almost unchanged as the number of attributes increases. Schemes [29, 36] use cloud servers and fog nodes for outsourcing, respectively, whereas edge nodes are used to undertake the outsourcing work in our scheme. Therefore, the encryption and decryption computation of our scheme is lower.

Figure 5(a) shows the time cost of token generation, and Figure 5(b) shows the search time. When the token is generated, the number of attributes is fixed to 5, and only the number of keywords changes. Figure 5(a) shows that the token generation time of our scheme is evidently lower than that in other schemes [31, 35]. Although the search time is constant in our scheme and scheme [35], the time cost in our scheme is lower. The time in scheme [31] increases with the number of attributes.

In conclusion, the experimental results are consistent with the theoretical analysis, and our scheme has better performance.

TABLE 2: Storage cost.

	PK	TPK	CT on DO	CT on CS/Fog/EN	Token
[29]	$2 U  G  +  G_T $	$(n+2) G $	$(2l+1) G  +  G_T  + 2 Z_p^* $	$l G $	—
[31]	$(2+ U ) G  +  G_T $	—	$(2d+2n+2) G  + 2 G_T $	—	$(2n+1) G  +  Z_p^* $
[35]	$4 G  +  G_T $	—	$(4+l) G  +  G_T $	—	$(n+6) G  +  Z_p^* $
[36]	$2 U  G  +  G_T $	$(n+1) G $	$2l G  + (l+1) G_T  + 3 Z_p^* $	$3l G $	—
Ours	$2 U  G  +  G_T $	$2 G $	$(1+d) G  +  G_T $	$2 G $	$(n_s+3) G $

TABLE 3: Computational cost.

	Encryption on DO	Encryption on CS/Fog/EN	Decryption on CS/Fog/EN	Decryption on DU	Token	Search
[29]	$(2l+1)E_G + E_P$	$2lE_G$	$(2n+1)E_P$	$E_P$	—	—
[31]	$(2d+2n+2)E_G + 2E_P$	—	—	—	$n_s(2n+1)E_G$	$E_G + (2n+1)E_P$
[35]	$(2l+5)E_G + E_P$	—	—	—	$n_s(n+7)E_G$	$6E_P$
[36]	$2lE_G + (l+1)E_P$	$3lE_G$	$(2n+4)E_P$	$E_P$	—	—
Ours	$(d+1)E_G + E_P$	$2E_G$	$3E_P$	$E_P$	$(n_s+3)E_G$	$4E_P$

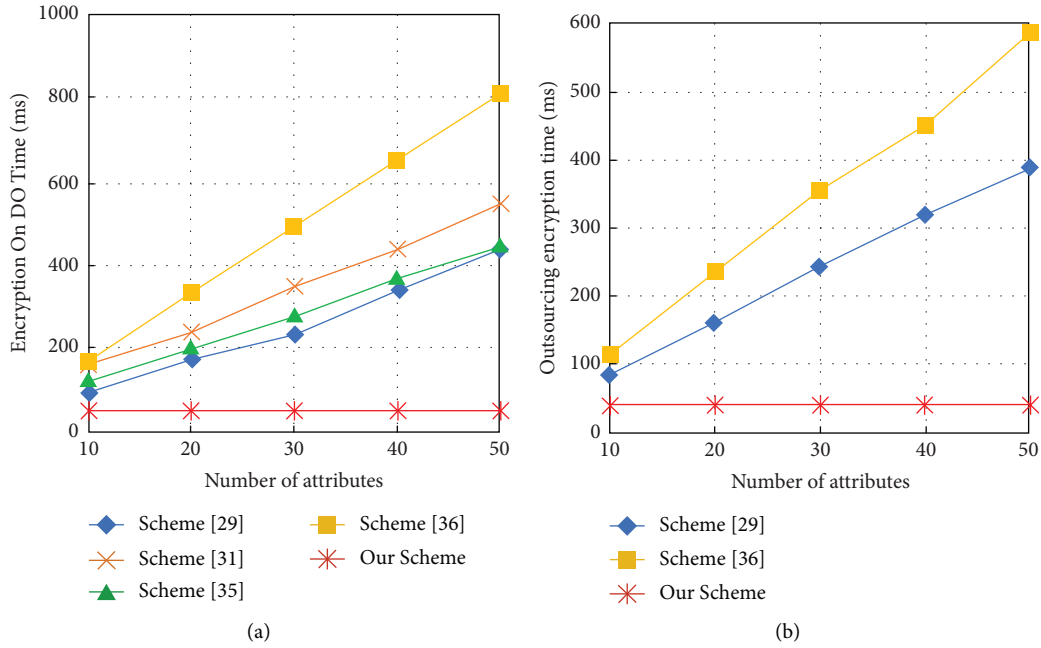


FIGURE 4: Continued.

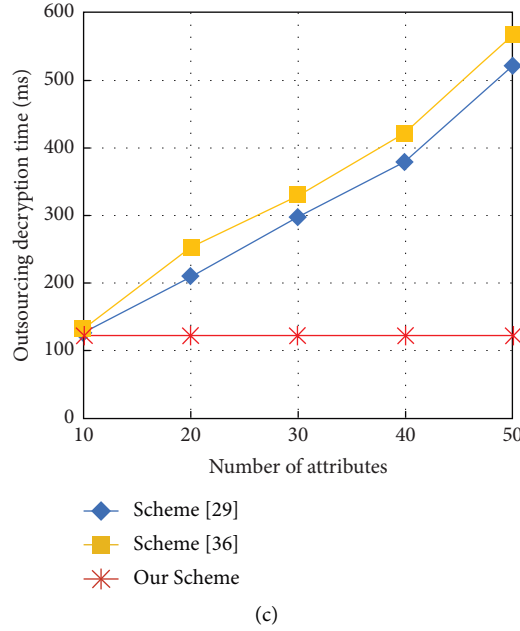


FIGURE 4: Time costs of DO encryption/outsourcing encryption/decryption phase. (a) DO enc. (b) Fog/CS/EN enc. (c) Fog/CS/EN dec.

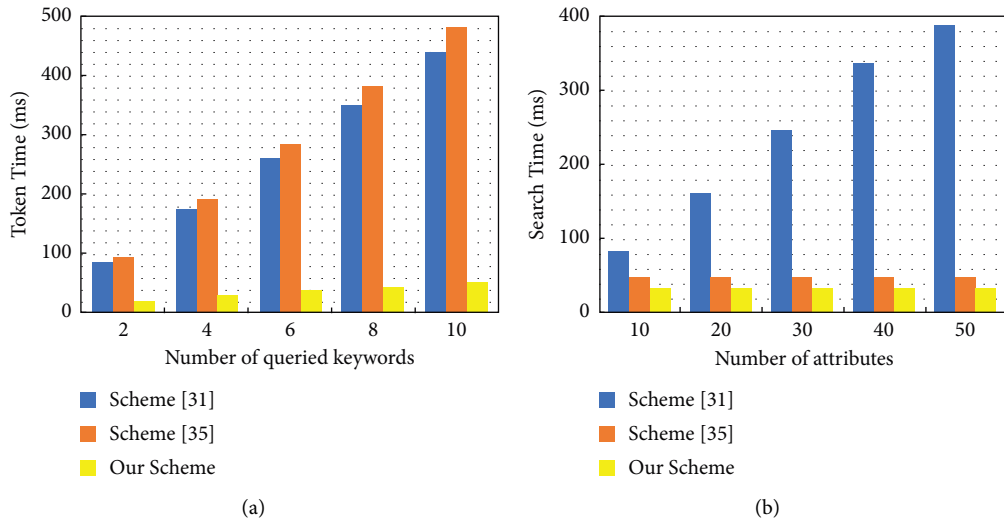


FIGURE 5: Time costs of token generation/search phase. (a) Token. (b) Search.

## 8. Conclusion

In the article, we present an ABKS scheme with multidata owner supporting policy updates. On the one hand, when encrypting shared messages, m-DOs adopt two-level access control, which realizes not only access control for multiple users through users' attribute sets and identities, but also the joint authorization management of multiple data owners, thereby improving the security of shared messages. On the contrary, based on the characteristics of low latency and low bandwidth operation, our scheme introduces edge computing. The edge node replaces the cloud server to complete the search task. As a result, the cloud cannot obtain keywords and other sensitive

information, and the delay caused by data transmission is reduced. When the access policy needs to be updated, m-DOs only send the updated access policy to EN, whereas index updating is undertaken by EN, greatly saving calculation and communication costs of DOs. We also outsourced some encryption and decryption calculations to edge nodes to reduce the computational burden of data owners and users. Theoretical analysis and experimental simulation show that the scheme is effective and practical in the edge computing environment.

As a part of our future work, we will continue to consider the hidden access policy to ensure the integrity and confidentiality of shared data and realize the secure storage of resources.

## Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

The work was supported by the National Natural Science Foundation of China (nos. 62262060 and 61662071) and the Industrial Support Plan Project of Gansu Provincial Department of Education (2022CYZC-17).

## References

- [1] C. P. Ge, W. Susilo, Z. Liu, J. Xia, P. Szalachowski, and F. Liming, "Secure keyword search and data sharing mechanism for cloud computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 6, pp. 1–2800, 2020.
- [2] J. Shen, T. Q. Zhou, and Z. F. Cao, "Protection methods for cloud data security," *Journal of Computer Research and Development*, vol. 58, no. 10, pp. 2079–2098, 2021.
- [3] D. Z. Liu, Y. Zhang, D. Jia, Q. S. Zhang, X. F. Zhao, and H. Rong, "Toward secure distributed data storage with error locating in blockchain enabled edge computing," *Computer Standards & Interfaces*, vol. 79, Article ID 103560, 2022.
- [4] Y. N. Li, Y. Yu, W. Susilo, Z. Y. Hong, and M. Guizani, "Security and privacy for edge intelligence in 5G and beyond networks: challenges and solutions," *IEEE Wireless Communications*, vol. 28, no. 2, pp. 63–69, 2021.
- [5] L. Zhang, H. Xiong, Q. Huang, J. G. Li, K. K. R. Choo, and J. T. Li, "Cryptographic solutions for cloud storage: challenges and research opportunities," *IEEE Transactions on Services Computing*, vol. 15, no. 1, pp. 567–587, 2022.
- [6] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Aarhus, Denmark, May 2005.
- [7] H. Y. Wang, Y. Li, W. Susilo, D. H. Duong, and F. Luo, "A fast and flexible attribute-based searchable encryption scheme supporting multi-search mechanism in cloud computing," *Computer Standards & Interfaces*, vol. 82, Article ID 103635, 2022.
- [8] Z. Liu, Y. Liu, J. Xu, and W. Baocang, "Verifiable attribute-based keyword search encryption with attribute revocation for electronic health record system," *International Journal on Network Security*, vol. 22, no. 5, pp. 845–856, 2020.
- [9] M. Ali, M. R. Sadeghi, X. M. Liu, Y. B. Miao, and A. V. Vasilakos, "Verifiable online/offline multi-keyword search for cloud-assisted Industrial Internet of Things," *Journal of Information Security and Applications*, vol. 65, Article ID 103101, 2022.
- [10] K. Zhang, Y. P. Li, and L. F. Lu, "Privacy-preserving attribute-based keyword search with traceability and revocation for cloud-assisted iot," *Security and Communication Networks*, vol. 2021, Article ID 9929663, 13 pages, 2021.
- [11] Y. B. Miao, R. H. Deng, K. K. R. Choo, X. M. Liu, J. T. Ning, and H. W. Li, "Optimized verifiable fine-grained keyword search in dynamic multi-owner settings," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 4, pp. 1–1820, 2019.
- [12] D. X. Song, W. David, and P. Adrian, "Practical Techniques for Searches on Encrypted Data," in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 44–55, Berkeley, CA, USA, May 2000.
- [13] D. Boneh, G. Di. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proceedings of the International Conference On the Theory And Applications Of Cryptographic Techniques*, pp. 506–522, Berlin, Heidelberg, May 2004.
- [14] D. J. Park, K. Kim, and P. J. Lee, "Public key encryption with conjunctive field keyword search," in *Proceedings of the International Workshop On Information Security Applications*, pp. 73–86, Springer, Berlin, Heidelberg, May 2004.
- [15] K. Kaushik, V. Varadharajan, and R. Nallusamy, "Multi-user attribute based searchable encryption," in *Proceedings of the 2013 IEEE 14th International conference on Mobile Data Management*, pp. 200–205, Milan, Italy, June 2013.
- [16] S. P. Wang, T. T. Gao, and Y. I. Zhang, "Searchable and revocable multi-data owner attribute-based encryption scheme with hidden policy in cloud storage," *PLoS One*, vol. 13, no. 11, Article ID 0206126, 2018.
- [17] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proceedings of the 20th USENIX conference on Security*, vol. 3, San Francisco, CA, USA, August 2011.
- [18] M. Ali and M. R. Sadeghi, "Provable secure lightweight attribute-based keyword search for cloud-based Internet of Things networks," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 5, Article ID e3905, 2021.
- [19] F. Meng, L. X. Cheng, and M. Q. Wang, "ABDKS: attribute-based encryption with dynamic keyword search in fog computing," *Frontiers of Computer Science*, vol. 15, no. 5, Article ID 155810, 2021.
- [20] Y. B. Miao, J. F. Ma, X. M. Liu, J. Weng, H. W. Li, and H. Li, "Lightweight fine-grained search over encrypted data in fog computing," *IEEE Transactions on Services Computing*, vol. 12, no. 5, pp. 772–785, 2019.
- [21] L. Sun, Z. Y. Zhao, J. H. Wang, and Z. Q. Zhu, "Attribute-based encryption scheme supporting attribute revocation in cloud storage environment," *Journal on Communications*, vol. 40, no. 5, pp. 47–56, 2019.
- [22] C. Payal and M. L. Das, "Keysea: Keyword-based search with receiver anonymity in attribute-based searchable encryption," *IEEE Transactions on Services Computing*, vol. 15, no. 213, 2020.
- [23] J. Ren, L. Y. Zhang, and B. C. Wang, "Decentralized multi-authority attribute-based searchable encryption scheme," *International Journal on Network Security*, vol. 23, no. 2, pp. 332–342, 2021.
- [24] X. Y. Liu, X. T. Yang, Y. K. Luo, L. Wang, and Q. Zhang, "Anonymous electronic health record sharing scheme based on decentralized hierarchical attribute-based encryption in cloud environment," *IEEE Access*, vol. 8, Article ID 200180, 2020.
- [25] A. Sahai, H. Seyalioglu, and B. Waters, "Dynamic credentials and ciphertext delegation for attribute-based encryption," in *Proceedings of the Annual Cryptology Conference*, pp. 199–217, Santa Barbara, CA, USA, August 2012.
- [26] J. Z. Lai, R. H. Deng, Y. J. Yang, and J. Weng, "Adaptable Ciphertext-Policy Attribute-Based Encryption," in *Proceedings of the International Conference On Pairing-Based Cryptography*, pp. 199–214, Beijing, China, August 2013.

- [27] Y. H. Z. Li, Z. Y. Dong, K. W. Sha, C. F. Jiang, J. Wan, and Y. Wang, "TMO: time domain outsourcing attribute-based encryption scheme for data acquisition in edge computing," *IEEE Access*, vol. 7, Article ID 40240, 2019.
- [28] K. Yang, X. H. Jia, K. Ren, R. T. Xie, and L. S. Huang, "Enabling efficient access control with dynamic policy updating for big data in the cloud," in *Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pp. 2013–2021, London, UK, May 2014.
- [29] Y. F. Tu, G. Yang, J. Wang, and Q. J. Su, "A secure, efficient and verifiable multimedia data sharing scheme in fog networking system," *Cluster Computing*, vol. 24, no. 1, pp. 225–247, 2021.
- [30] X. X. Yan, Y. Liu, Z. C. Li, and Y. L. Tang, "Tabe-D A C: Multi-authority attribute-based encryption scheme with policy dynamic updating," *Journal on Communications*, vol. 38, no. 10, 2017.
- [31] Y. B. Miao, X. M. Liu, K. K. R. Choo et al., "Privacy-preserving attribute-based keyword search in shared multi-owner setting," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1080–1094, 2021.
- [32] A. Beimel, "Secure schemes for secret sharing and key distribution," 1996, <https://www.cs.bgu.ac.il/%7Ebeimel/Papers/thesis.pdf>.
- [33] S. S. M. Chow, "Removing escrow from identity-based encryption," in *Proceedings of the International Workshop on Public Key Cryptography*, pp. 256–276, Irvine, CA, USA, May 2009.
- [34] S. Song and X. L. Zhang, "Attribute-based encryption scheme without key escrow supporting attribute revocation in cloud environment," *Netinfo Security*, vol. 20, no. 8, pp. 62–70, 2020.
- [35] M. S. Cao, L. H. Wang, Z. G. Qin, and C. W. Lou, "A lightweight fine-grained search scheme over encrypted data in cloud-assisted wireless body area networks," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 9340808, 12 pages, 2019.
- [36] K. Fan, T. T. Liu, K. Zhang, H. Li, and Y. Yang, "A secure and efficient outsourced computation on data sharing scheme for privacy computing," *Journal of Parallel and Distributed Computing*, vol. 135, pp. 169–176, 2020.
- [37] L. Ben, "PBC library manual," 2006, <https://crypto.stanford.edu/pbc/manual/>.